ON AN ADAPTIVE ALGORITHM FOR SOLVING INITIAL VALUE PROBLEMS OF O.D.E. AND ITS EFFECTIVENESS(Numerical Analysis and their Algorithms)

AUTHOR(S):

Wang-yao, Li

CITATION:

Wang-yao, Li. ON AN ADAPTIVE ALGORITHM FOR SOLVING INITIAL VALUE PROBLEMS OF O.D.E. AND ITS EFFECTIVENESS(Numerical Analysis and their Algorithms). 数理解析研究所講究録 1992, 791: 100-109

ISSUE DATE:

1992-06

URL:

http://hdl.handle.net/2433/82689

数理解析研究所講究録
第 791 巻 1992 年 100-109

ON AN ADAPTIVE ALGORITHM FOR SOLVING INITIAL VALUE PROBLEMS OF O. D. E. AND ITS EFFECTIVENESS

Li Wang–yao
(*Computing Center, Academia Sinica, Beijing*)

Abstract

In this paper, we introduce an adaptive algorithm for solving initial value problems of O. D. E. and the numerical tests show that it is dominant in comparison with Gear's automatic code.

1 Introduction

Algorithms with fixed order and fixed stepsize have long been used to solve initial value problems of ordinary differential equations, but when an answer with definite accuracy and smallest amount at work is required, as is often the case, those algorithms cannot work very effectively and satisfy the demand. Generally, the change rate of functions to be solved varies during the whole process. A desirable accuracy can be obtained with a small stepsize in a fast variation region, but it will be more expensive and useless to take the same stepsize in a slow variation region because locally higher accuracy is unprofitable to the user. Theory and practical experiences have indicated that methods with high order are suitable for high accuracy problems and those with low order are economical for low accuracy problems. Therefore, algorithms with automatic control of stepsize and order are desired, and to this end, we need to estimate the local truncation error for one-step methods in a convenient and economical manner. For multistep methods, convenient and economical transformation among three methods with neighbouring orders, besides their local truncation errors, is also demanded. Of course, we also hope that the stepsize can be changed easily.

In late 1960s, C. W. Gear successfully solved this problem by use of the Nordsieck notation and presented a general-purpose automatic program for the Adams-Moulton method and the BDF formula. T.E. Hull([1,2]) highly praised this program: "If a program library was to contain only one program for solving ordinary differential equations, we would strongly recommend Gear's."

Soon, various programs with automatically changing stepsize were presented for Runge-Kutta methods. Because no techniques are currently available for selecting the order in Runge-Kutta methods, those programs are of fixed order.

In early 1960s, new problems – stiff equations – arose in various fields such as chemical kinetics, automatic control, and network analysis, various methods were proposed, such as Gear's methods, Enright's method, the method based on the trapezoidal rule with extrapolation, implicit Runge-Kutta methods for stiff problems. Up to now, however some of problems are still not solved.

For a stiff problem, we must choose a small stepsize to gain a definite accuracy due to error tolerance in a fast variation region (i.e. transiency). Here the Adams methods are probably more advantageous than the BDF methods. Although we need to choose a smaller stepsize to

ensure the stability and convergence of the simple iteration for the Adams methods, the total cost probably is less than that of the BDF methods as the latter Newton iteration inflicts a huge cost. This situation is called the nonstiff stage of stiff problems. Obviously, the Adams-Moulton method is more economical in the nonstiff stage. Moreover since the eigenvalues of the system vary in size for nonlinear stiff problems, the solution of the problems sometimes is stiff and sometimes is nonstiff. In this case it is not economical to use the Adams or the BDF method alone. The best treatment is automatic change of numerical integration methods according to the properties of the solution.

The problems with discontinuity of right function often appear in some practical problems such as transitional input in control systems. When integrating these problems in automatic code, the mistakes of alternate judgment in the neighbourhood of discontinuity arise so often that the amount of computation increases greatly. Therefore, an automatic program which can pass a discontinuous point automatically and efficiently is also desired.

The automatic program that automatically changes stepsize, order and integration methods and passes discontinuity arose in early 1980's ([3]). It is a great progress in applying adaptive algorithms to solving initial value problems of ordinary differential equations.

Even if the users do not know any information about the problems to be solved, such as whether the problem is stiff or not or whether the right functions have discontinuity or not, this program can still provide a good answer reasonably and efficiently. It can also solve stiff problems more efficiently.

Now we introduce the principle, the criterion and the main points for implementing the Adams–BDF automatic program as follows.

2 Automatic Control of Stepsize and Order

In automatic control of stepsize and order, after finishing each integration step we should check, as a principle, if the local truncation error is smaller than the error tolerance required by the user.

For the Adams–Moulton method (or the BDF method) we require

$$c_{k+1}y^{(k+1)}h^{(k+1)} \leq \varepsilon \tag{1}$$

where c_{k+1} is the error constant, h is the integration stepsize and ε is error tolerance. If the relationship (1) is satisfied, we accept this integration step (it is called a successful step). At the same time we estimate such an H, that satisfies (1) for the methods with order $k-1$, k, and $k+1$ respectively, select the largest H from them as the stepsize in the next step, and change the current order of the method corresponding to this H.

For a faulty step, an analogous judgment is made to define a new stepsize, which is used for recalculation of this step. The detail of the control strategy is described in [4].

We notice that only $y^{(k+1)}$ is unknown in (1). Of course, $y^{(k)}$ and $y^{(k+2)}$ are also required in order to change the order. The approximate values of these unknowns may be obtained by interpolation. For example, for the BDF method with order k, $y_n^{(k)}$ may be obtained through $k+1$ known values of the function and derivative $y_n, hy'_n, y_{n-1}, \cdots, y_{n-k+1}$, by interpolation. Meanwhile $y_n^{(k+1)}$ and $y_n^{(k+2)}$ can be obtained through $\nabla y_n^{(k)}$ and $\nabla^2 y^{(k)}$.

When the stepsize is changed, the values at each network point relative to the new stepsize may be calculated in terms of the obtained values of the functions and derivatives.

There is no difficulty in principle, but the numerical work is huge and cannot be borne due to repeated interpolation.

C. W. Gear successfully solved this difficult problem using the Nordsieck notation. The BDF formula with order $k, y_n = \sum_1^k \alpha_i y_{n-i} + h\beta_0 f_n$, can be transferred into an equivalent predictor-corrector formula

$$\begin{cases} a_n^{(0)} = Ba_{n-1} \\ a_n^{(i+1)} = a_n^{(i)} + LF(a_n^{(i)}) \end{cases} \tag{2}$$

Where

$$\begin{aligned} a_n &= (a_{on}, a_{1n} \cdots a_{kn})^T = (y_n, hy'_n, \frac{h^2}{2!}y_n^{(2)} \cdots \frac{h^k}{k!}y_n^{(k)})^T \\ &= A(y_n, hy'_n, y_{n-1} \cdots y_{n-k+1})^T \end{aligned}$$

B is the Pascal triangular matrix with order $k+1$, $L = (L_0 \cdots L_k)^{-1}$ is a constant vector, $F(a_n) = hf(a_{0n}) - a_{1n}$ and A is the matrix constituted by the coefficients of the interpolation polynomial. In this way, it is very easy to obtain the derivatives $y_n^{(k)}, y_n^{(k+1)}$ and $y_n^{(k+2)}$ using (2). Furthermore, changing stepsize vecomes very easy, like one-step methods, and so does the order change. It is obvious that the BDF and the Adams-Moulton formula have the same form of predictor-corrector formula except that the vector L is different. It makes the changing integration methods easy to implement.

3 Automatical Change of Integration Methods

The main principles of changing stepsize, order and integration methods are as follows; when the order of the methods, the stepsize and the integration methods need changing, we separately estimate the stepsize H for the BDF and the Adams-moulton methods with order $k-1, k$ and $k+1$ according to the criterion $c_{K+1}h^{k+1}y^{(k+1)}$ <Eqs, and obtain two groups of $H, h_{B,k-1}, h_{B,k}, h_{B,k+1}$ (for BDF) and $h_{A,k-1}, h_{A,k}, h_{A,k+1}$ (for Adams). It is impossible to ensure the stability and effectiveness of the calculation for the Adams-Moulton methods using such an H determined by the tolerance. Therefore, the restrictions on the stability and convergence of the simple iteration must be considered.

According to the restriction of stability

$$h_{A,k}^* \leq \frac{\text{Disks } (k)}{\text{ESTL}} \tag{3}$$

We obtain a group of $H, h_{A,k-1}^*, h_{A,k}^*, h_{A,k+1}^*$, once more.

In (3), ESTL $= \sqrt{\dfrac{\sum_i (f_i^{n+1} - f_i^n)^2}{\sum_i (y_i^{n+1} - y_i^n)^2}}$ is an approximate Lipschitz constant, where the superscripts n and $n+1$ denote neighbouring values formed in correction interaction, Disks (k)

denotes the radius of the maximal semidisk, which is almost covered by the stability region of the method with order k. The Disks (k) of the Adams-Moulton methods are listed in Table 1.

When the functional equation $y = h\beta_0 f(y) + C$ is solved by means of simple iteration, the iterative formula is $\dfrac{y^{n+1} - y^n}{y^n - y^{n-1}} = h\beta_0 \dfrac{\partial f}{\partial y}$. If we expect the iterative rate to be larger than 2, then $h * \text{ESTL} < \dfrac{1}{2\beta_{k0}}$, where $\beta_{k,0}$ is the coefficient of the Adams-Moulton formula. We let $\dfrac{1}{2\beta_{k,0}} = \text{Disks } (k)$ and call this Disks(k) the radius of the convergence disk and list them Table 2. If fewer values of Disks (k) in Table 1 and Table 2 are chosen (as listed in Table 3) and used in relationship (3), when H satisfied relationship (3), the restrictions on the stability and convergence of the simple iteration are satisfied simultaneously.

Choosing $\bar{h}_{A,k} = \min(h_{A,k}, h^*_{A,k})$ we obtain $\bar{h}_{A,k-1}, \bar{h}_{A,k}$ and $\bar{h}_{A,k+1}$. Such a group of $\bar{h}_A$ satisfies the restriction on accuracy, stability, and convergence of the simple iteration at the same time, At last, we choose $h = \max(h_{B,k-1}, h_{B,k}, h_{B,k+1}, \bar{h}_{A,k-1}, \bar{h}_{A,k}, \bar{h}_{A,k+1},)$ as the predictive value of the next stepsize and at the same time, a new order and new integration methods are determined.

Due to the different costs between the Adams-Moulton method and the BDF method in one step integration, when we judge whether the Adams or the BDF method should be adopted, a weighting factor must be added. If $\bar{h}_{A,k} \geq \dfrac{\text{SADM}}{\text{SBDF}} \cdot h_{B,k}$, then we choose $\bar{h}_{A,k}$ otherwise we choose $h_{B,k}$. Here, $\dfrac{\text{SADM}}{\text{SBDF}}$, the weighting factor, is the ratio of the calculation work of the Adams method to that of BDF method in one step integration.

Automatical passing of the discontinuous point of right functions is described in [5].

As stated above, all criteria for changing order, stepsize and integration methods are obtained during the numerical process. Therefore, a large amount of additional work is avoided. Using the Nordsieck notation makes the process of changing order, stepsize and integration methods very simple, giving the automatic program great vitality.

4 Numerical Tests

In order to identify the effectiveness of the "AS" for solving stiff problems in O.D.Es, seventy-two stiff problems adopted by Enright, etc., were solved. The numerical tests show that "AS" is dominant in comparison with Gear's automatic code.

(1) SOME EXPLANATION

The calculations were implemented in IBM-AT using double precision.

CODE: "AS" present adaptive solver INNER1 in ODE package [6]. "GS" present Gear's automatic code DIFSUB [4].

PROBLEMS: Five classes of stiff problems (note A,B,C,D and E class respectively, except E4) adopted by Enright, etc. [2], were solved.

A class contains four linear stiff problems with real eigenvalues.

B class contains five linear stiff problems with non real eigenvalues.

C class contains five non linear stiff problems coupling.

D class contains five non linear stiff problems with real eigenvalues.

E class contains four non linear stiff problems with non real eigenvalues.

Each problem was calculated using three kinds of error tolerance (10^{-2}, 10^{-4}, 10^{-6}) respectively, so all of test problems total seventy - two.

Above stiff problems appear in wide field of science and technology such as physics, chemistry, insulator physics, control theory, nuclear reactor theory, reactor kinetics, circuit theory etc.

ERROR CONTROL: Relative error control

INITIAL STEPSIZE: The initial stepsize are given by the code automatically for "AS," but reasonable initial stepsizes are predetermined for "GS."

STATISTICS: The following statistics were chosen to reflect cost.

T: The total time required to solve a problem. It mainly reflects auxiliary calculation times because all of the test problems are of small size (the orders of differential equations are smaller then ten and evaluation of functions and Jacobian are vary simple).

FN: The number of function evaluations

JN: The number of Jacobian evaluations

IN: The number of matrix inversions

SN: The number of integration steps

(2) COMPUTATION RESULT

It is listed in the following statistic table 4.

(3) Conclusion

(1) "AS" is overall dominant in comparison with "GS" when calculations proceed with low precision (error tolerance 10^{-2}).

(2) "AS" is almost overall dominant in comparison with "GS" for non linear stiff problems C,D and E classes

(3) "GS" failed for problems B5 and E5, but "AS" was successful.

(4) "GS" is slightly better then "AS," when calculations proceed with middle precision (error tolerance 10^{-4}) for A class problems.

(5) The auxiliary calculation time of "AS" is great then "GS" due to judgment and comparisons increase in code of "AS," but it will be less important with increase of size of problems to be solved

Table 1

k	1	2	3	4	5	6	7	8	9	10
Disks	∞	∞	1	1	1.25	1.1	0.7	0.45	0.25	0.15

Table 2

k	1	2	3	4	5	6	7	8	9	10
Disks	0.5	1	1.2	1.3	1.43	1.5	1.58	1.64	1.69	1.74

Table 3

k	1	2	3	4	5	6	7	8	9	10
Disks	0.5	1	1	1	1.25	1.1	0.7	0.45	0.25	0.15

References

[1] T. E. Hull, W. H. Enright, B. M. Fellen and A. E. Sedgwick, Comparing numerical methods for ordinary differential equations, *SIAM J. Numer. Anal.*, **9** : 4 (1972), 603–637.

[2] W. H. Enright, T. E. Hull and B. Lindberg, Comparing numerical methods for stiff systems of O. D. Es, *BIT,* **15** (1975), 10–48.

[3] Software Package for Numerical Integration of O. D. Es, Department of Computer Science, University of Illinois.

[4] C. W. Gear, Numerical Initial Value Problems in Ordinary Differential Equations, Prentice-Hall, Englewood cliffs, New Jersey, 1971.

[5] C. W. Gear, O. Østerby, Solving ordinary differential equations with discontinuities, REPT. No. UIUCDCS-R-81-1064.

[6] Li Wang-yao and Mao Zu-fan, Software package for initial value problems of O. D. Es, Computing Center, Academia Sinica.

Table 4 – 1

Tolerance	problem	method	T	F_n	J_n	I_n	S_n
10^{-2}	a1	as	5	93	5	5	37
10^{-2}	a1	gs	4	93	10	10	40
10^{-2}	a2	as	12	129	10	10	52
10^{-2}	a2	gs	12	118	17	17	49
10^{-2}	a3	as	7	149	10	10	56
10^{-2}	a3	gs	5	144	15	15	58
10^{-2}	a4	as	22	239	15	15	88
10^{-2}	a4	gs	18	195	21	21	73
10^{-4}	a1	as	11	295	21	21	107
10^{-4}	a1	gs	6	224	15	15	94
10^{-4}	a2	as	26	335	26	26	124
10^{-4}	a2	gs	22	333	21	21	121
10^{-4}	a3	as	12	337	29	29	123
10^{-4}	a3	gs	7	329	20	20	129
10^{-4}	a4	as	50	575	38	38	220
10^{-4}	a4	gs	31	435	23	23	153
10^{-6}	a1	as	15	383	7	7	161
10^{-6}	a1	gs	10	458	18	18	186
10^{-6}	a2	as	52	661	22	22	279
10^{-6}	a2	gs	41	613	35	35	245
10^{-6}	a3	as	22	647	36	36	246
10^{-6}	a3	gs	14	602	33	33	244
10^{-6}	a4	as	93	1129	71	71	434
10^{-6}	a4	gs	56	702	41	41	293
10^{-2}	b1	as	10	259	15	15	110
10^{-2}	b1	gs	16	758	17	17	304
10^{-2}	b2	as	6	89	3	3	39
10^{-2}	b2	gs	5	89	10	10	37
10^{-2}	b3	as	7	111	10	10	42
10^{-2}	b3	gs	6	103	11	11	42
10^{-2}	b4	as	8	147	12	12	56
10^{-2}	b4	gs	7	153	16	16	64
10^{-2}	b5	as	14	275	13	13	114
10^{-2}	b5	gs	207	7001	20	20	2549

Table 4 – 2

Tolerance	problem	method	T	F_n	J_n	I_n	S_n
10^{-4}	b1	as	29	769	24	24	344
10^{-4}	b1	gs	21	1036	20	20	407
10^{-4}	b2	as	12	219	7	7	87
10^{-4}	b2	gs	10	241	15	15	102
10^{-4}	b3	as	13	259	11	11	104
10^{-4}	b3	gs	9	229	13	13	90
10^{-4}	b4	as	18	373	20	20	146
10^{-4}	b4	gs	13	335	19	19	127
10^{-4}	b5	as	30	655	37	37	258
10^{-4}	b5	gs	213	6876	33	33	2677
10^{-6}	b1	as	65	1719	28	28	802
10^{-6}	b1	gs	35	1827	32	32	699
10^{-6}	b2	as	24	499	12	12	209
10^{-6}	b2	gs	13	354	15	15	142
10^{-6}	b3	as	23	499	12	12	224
10^{-6}	b3	gs	29	778	9	9	355
10^{-6}	b4	as	39	887	35	35	343
10^{-6}	b4	gs	20	590	27	27	228
10^{-6}	b5	as	56	1189	24	24	511
10^{-6}	b5	gs	289	7489	29	29	2848
10^{-2}	c1	as	5	123	5	5	51
10^{-2}	c1	gs	5	132	12	12	50
10^{-2}	c2	as	5	111	4	4	45
10^{-2}	c2	gs	4	127	11	11	46
10^{-2}	c3	as	6	115	4	4	45
10^{-2}	c3	gs	4	116	14	14	42
10^{-2}	c4	as	6	135	14	14	51
10^{-2}	c4	gs	4	129	16	16	47
10^{-2}	c5	as	6	135	12	12	48
10^{-2}	c5	gs	6	188	19	19	51
10^{-4}	c1	as	12	307	14	14	116
10^{-4}	c1	gs	7	301	14	14	111
10^{-4}	c2	as	10	271	23	23	100
10^{-4}	c2	gs	7	279	13	13	104

Table 4 – 3

Tolerance	problem	method	T	F_n	J_n	I_n	S_n
10^{-4}	c3	as	10	277	18	18	102
10^{-4}	c3	gs	7	278	16	16	102
10^{-4}	c4	as	12	301	24	24	108
10^{-4}	c4	gs	7	296	19	19	105
10^{-4}	c5	as	11	216	20	20	98
10^{-4}	c5	gs	9	387	31	31	108
10^{-6}	c1	as	19	499	9	9	220
10^{-6}	c1	gs	13	544	23	23	211
10^{-6}	c2	as	18	489	12	12	192
10^{-6}	c2	gs	12	466	25	25	188
10^{-6}	c3	as	19	529	20	20	196
10^{-6}	c3	gs	12	502	23	23	196
10^{-6}	c4	as	1 9	457	15	15	194
10^{-6}	c4	gs	13	531	27	27	193
10^{-6}	c5	as	21	533	20	20	218
10^{-6}	c5	gs	14	623	30	30	198
10^{-2}	d1	as	3	95	7	7	35
10^{-2}	d1	gs	2	124	21	21	30
10^{-2}	d2	as	4	97	12	12	36
10^{-2}	d2	gs	3	109	15	15	40
10^{-2}	d3	as	5	111	5	5	51
10^{-2}	d3	gs	5	138	19	19	51
10^{-2}	d4	as	3	29	4	4	10
10^{-2}	d4	gs	2	32	6	6	11
10^{-2}	d5	as	3	79	14	14	21
10^{-2}	d5	gs	2	74	15	15	21
10^{-2}	d6	as	3	43	7	7	14
10^{-2}	d6	gs	3	36	7	7	14
10^{-4}	d1	as	5	203	12	12	77
10^{-4}	d1	gs	3	284	28	28	79
10^{-4}	d2	as	8	221	11	11	89
10^{-4}	d2	gs	5	243	20	20	92
10^{-4}	d3	as	13	307	9	9	141
10^{-4}	d3	gs	8	335	17	17	137
10^{-4}	d4	as	3	41	4	4	16
10^{-4}	d4	gs	2	45	8	8	18
10^{-4}	d5	as	4	211	17	17	61
10^{-4}	d5	gs	3	160	20	20	39

Table 4 – 4

Tolerance	problem	method	T	F_n	J_n	I_n	S_n
10^{-4}	d6	as	4	95	11	11	32
10^{-4}	d6	gs	3	93	9	9	34
10^{-6}	d1	as	9	387	32	32	144
10^{-6}	d1	gs	5	391	42	42	116
10^{-6}	d2	as	12	379	8	8	161
10^{-6}	d2	gs	7	396	23	23	157
10^{-6}	d3	as	21	527	9	9	236
10^{-6}	d3	gs	16	659	25	25	267
10^{-6}	d4	as	4	93	11	11	35
10^{-6}	d4	gs	3	79	8	8	31
10^{-6}	d5	as	7	311	20	20	107
10^{-6}	d5	gs	4	338	29	29	100
10^{-6}	d6	as	7	155	12	12	60
10^{-6}	d6	gs	4	146	11	11	59
10^{-2}	e1	as	3	23	4	4	7
10^{-2}	e1	gs	3	30	4	4	11
10^{-2}	e2	as	2	31	4	4	11
10^{-2}	e2	gs	2	24	5	5	13
10^{-2}	e3	as	4	105	4	4	38
10^{-2}	e3	gs	3	100	12	12	37
10^{-2}	e5	as	3	35	4	4	13
10^{-2}	e5	gs	4	100	19	19	28
10^{-4}	e1	as	4	77	12	12	26
10^{-4}	e1	gs	4	85	11	11	27
10^{-4}	e2	as	3	71	4	4	25
10^{-4}	e2	gs	2	80	6	6	32
10^{-4}	e3	as	8	207	13	13	80
10^{-4}	e3	gs	5	231	18	18	87
10^{-4}	e5	as	4	43	5	5	16
10^{-4}	e5	gs	failure				
10^{-6}	e1	as	7	157	4	4	59
10^{-6}	e1	gs	7	194	14	14	78
10^{-6}	e2	as	3	115	0	0	48
10^{-6}	e2	gs	3	154	9	9	61
10^{-6}	e3	as	15	497	26	26	189
10^{-6}	e3	gs	8	428	24	24	161
10^{-6}	e5	as	4	73	10	10	24
10^{-6}	e5	gs	failure				